VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2019 Jakub Širjov

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

LÉTAJÍCÍ PLATFORMA ZALOŽENÁ NA OPERAČNÍM SYSTÉMU FLYTOS

AERIAL PLATFORM BUILT ON OPERATING SYSTEM FLYTOS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Jakub Širjov**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jiří Pokorný**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Jakub Širjov ***ID:*** 197644
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Létající platforma založená na operačním systému FlytOS

POKYNY PRO VYPRACOVÁNÍ:

Cílem bakalářské práce bude realizovat létající platformu založenou na operačním systému FlytOS. Bude vybrán vhodný hardware pro řídicí jednotku, na které FlytOS bude bude spuštěn. Platforma bude uvažována pro využití jako létající přístupový bod využívající některou z bezdrátových technologií (Wi-Fi, LTE). Bude nutné řešit horizontální rotaci platformy z důvodu udržení páteřního spoje se základnovou stanicí. Bude navržen scénář automatizovaného letu pro poskytnutí připojení uživatelům ve vzdáleném místě. Ten bude nejprve odsimulován v simulačním prostředí FlytSIM. Po úspěšném odsimulování testovacího letu bude proveden let v reálném prostředí.

DOPORUČENÁ LITERATURA:

[1] SADRAEY, Mohammad. Unmanned Aircraft Design: A Review of Fundamentals. Synthesis Lectures on Mechanical Engineering, 2017, 1.2: i-193.

[2] CHEN, Hai; WANG, Xin-min; LI, Yan. A survey of autonomous control for UAV. In: Artificial Intelligence and Computational Intelligence, 2009. AICI'09. International Conference on. IEEE, 2009. p. 267-271.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Jiří Pokorný
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Bakalárska práca je zameraná na výber a zostrojenie funkčného modelu lietajúcej platformy, ktorá musí byť schopná uniesť pridané zariadenia. Lietajúca platforma má slúžiť na vytvorenie prístupového bodu pre určitú oblasť, pričom musí celý čas udržiavať spojenie so základovou stanicou. Na lietajúcu platformu je nainštalovaný FlytOS, ktorý slúži na ovládanie lietajúcej platformy. Ďalšia časť práce popisuje kroky spojené so simuláciou prvého automatizovaného letu s nastaveným parametrom otáčania v prostredí FlytSIM. Následne je overené správne fungovanie tejto platformy pomocou manuálneho ovládania diaľkovým ovládačom a v závere tejto časti je otestovaný simulovaný automatizovaný let v reálnych podmienkach. Záverečná časť práce sa zaoberá algoritmom, ktorý na základe vstupných údajov vypočíta uhol natočenia k základovej stanici. V prípade veľkých vzdialenosti medzi bodmi letu vypočíta medzisúradnice a ich natočenie.

KĽÚČOVÉ SLOVÁ

UAV, dron, FlytOS, automatizovaný let, multikoptéra, bezpilotné lietajúce vozidlo, lietajúci prístupový bod

ABSTRACT

The aim of this bachelor's work is creation of functional model of flying platform that can carry additional devices. The platform is designed to maintain continuous connection with the base station and to create an access point for a particular area. The flying platform has FlyOS installed, which facilitates control of the flying platform. Another part of the work describes steps for simulation of the first automated flight in FlytSIM with a preset parameter of rotation during flight. The platform's functionality is then verified by using manual remote control and also by simulated automated flight in real conditions. Final part shifts attention to the algorithm used for the computation of the rotation of the flight waypoints direction in degrees towards the base station based on the input data. In the case of large distances between the flight waypoints, it computes coordinates and the rotation of the UAV.

KEYWORDS

UAV, drone, FlytOS, automated flight, multicopter, Unmanned Aerial Vehicle, flying access point

ŠIRJOV, Jakub. *Létající platforma založená na operačním systému FlytOS*. Brno, 2019, 48 s. Bakalárska práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: Ing. Jiří Pokorný

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Létající platforma založená na operačním systému FlytOS“ vypracoval samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Jiřímu Pokornému za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno ....

podpis autora

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic
http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výzkum popsaný v tejto bakalárskej práci bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

Brno ....

podpis autora

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

Obsah

Zoznam obrázkov

Úvod

Názov „dron“ sa u nás začal používať v súvislosti s prebratým hovorovým anglickým slovom „drone“. Je to lietadlo, ktoré sa spisovne označuje Unmanned Aerial Vehicle, skrátene UAV a to znamená bezpilotné lietadlo. UAV je riadené na diaľku, alebo môže lietať samostatne pomocou predprogramovaných letových plánov, prípadne pomocou zložitejších autonómnych subsystémov. V praxi taktiež existujú označenia ako UA (Unmanned aircraft) a RPA (Remotely piloted aircraft). Slovné spojenie Autonomous aircraft znamená autonómny bezpilotný prostriedok, ktorý nedovoľuje zásah pilota do nastaveného letu zariadenia [1]. Vývoj bezpilotných lietadiel realizovala armáda, ktorej snahou bolo znížiť straty na životoch počas bojových operácií.

Teoretická časť bakalárskej práce sa zaoberá výberom vhodného rámu kvadrokoptéry a metódy jej zostrojenia. Následne sa zameriava na výber vhodných súčiastok, ktoré nebudú energeticky náročné, nakoľko kvadrokoptéra bude slúžiť na dlhý pokojný let. Taktiež musí byť schopná lietať s potenciálnou záťažou, ktorou by mala byť anténa na komunikáciu s pozemnou stanicou. Vyberané súčiastky sú motory, vrtule, akumulátor a radiče. Jej rám bude vyberaný v závislosti na požiadavkách ako sú cena, dostupnosť a jednoduchosť výroby. Záverečné kapitoly teoretickej časti popisujú softvér použitý v tejto práci, a to FlytOS, Ground Control Station a FlytSIM.

Začiatok praktickej časti je zameraný na vybrané súčiastky a tiež na samotnú výrobu a zostrojenie tela kvadrokoptéry. Následne je popísaný celý postup inštalácie FlytOS na zariadenie Raspberry Pi. Toto zariadenie slúži pre ovládanie kvadrokoptéry prostredníctvom riadiacej jednotky Pixhawk. V praktickej časti je následne zobrazené fyzické prepojenie Raspberry Pi a riadiacej jednotky Pixhawk. Ďalej je vysvetlené, akými metódami je možné vytvoriť automatizovaný let v prostredí FlytSIM, taktiež je aj znázornený prvý odsimulovaný automatizovaný let UAV a vypísaný kód pre jeho let. Dôležitou častou je zostrojenie celej kvadrokoptéry a overenie jej funkčnosti a letových schopností prostredníctvom diaľkového ovládača. Následne je otestovaný prvý automatizovaný let zo simulácie v reálnych podmienkach a kvadrokoptéra by mala počas celého letu udržiavať natočenie na základovú stanicu. V budúcnosti by mohlo UAV poskytovať prístupový bod pre určitú oblasť, kde nie je dobrý prístup a preto musí byť počas celého letu neustále nasmerovaná na základovú stanicu. Koniec praktickej časti sa zaoberá algoritmom pre vytvorenie automatizovaného letu, do ktorého budú zadané vstupné súradnice a dodatočné údaje. Výstupom algoritmu bude súbor v jazyku Python s automatizovaným letom, kde budú všetky súradnice nasmerované na základovú stanicu a tento súbor dokáže spracovať FlytOS. Tento automatizovaný let bude overený v prostredí FlytSIM.

1 Teoretická časť

Multikoptéra je viac vrtuľová platforma. Je schopná vertikálne vzlietnuť a pristáť, udržať sa na mieste vo vzduchu a lietať rôznymi smermi. Predpona „multi“ znamená väčší počet. Podľa počtu rotorov sa odvodzuje názov tak, že k základu slova sa na miesto slova multi pridáva číselná predpona. Predmetom práce je multikoptéra so štyrmi rotormi, ktorá sa nazýva kvadrokoptéra.

Z technického hľadiska sú modely UAV vyrobené z ľahkých materiálov. Musia byť odolné voči vonkajším vplyvom, ako napríklad vietor, slnko a dážď. Zjednodušene platí, že čím je menšia hmotnosť UAV, tým rýchlejšie vzlietne a má nižšiu spotrebu energie [2].

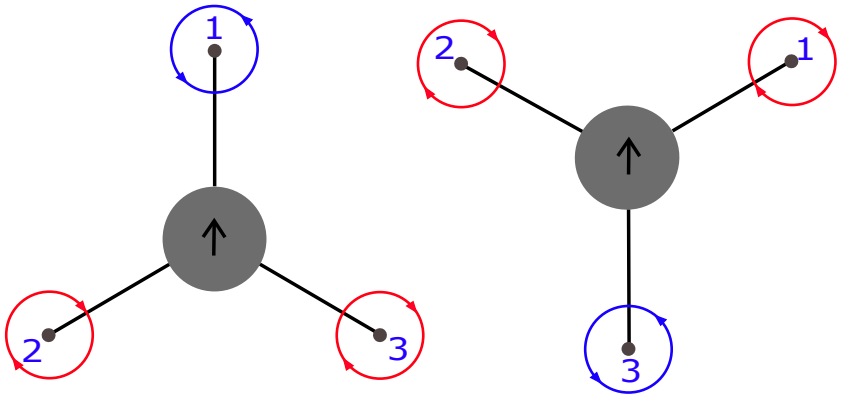
Pre ovládanie UAV sa často využíva rádiové spojenie. Rádiové vlny na komunikáciu medzi pilotom a UAV postupne nahradili moderné dátové siete, ako napríklad Wi-Fi. Tieto siete umožňujú za krátky čas rýchly prenos veľkého množstva informácií, avšak ich nevýhodou je vyššia spotreba energie [3] [2].

1.1 Druhy multikoptér

Množstvo motorov na UAV ma zásadný vplyv na jeho letové schopnosti. Z konštrukčného hľadiska delíme UAV podľa počtu ramien na platforme. Sú to trikoptéry, kvadrokoptéry, hexakoptéry a oktakoptéry. Všeobecne sa dá povedať, že čím menej má multikoptéra ramien, tým má väčšiu obratnosť. Vyšší počet ramien zase pridáva na stabilite [4].

1.1.1 Trikoptéry

Je to UAV s tromi ramenami. Veľkou výhodou tohto variantu je vysoká obratnosť, preto sú využívané hlavne na akrobatické účely. Negatívnymi vlastnosťami tejto plat-

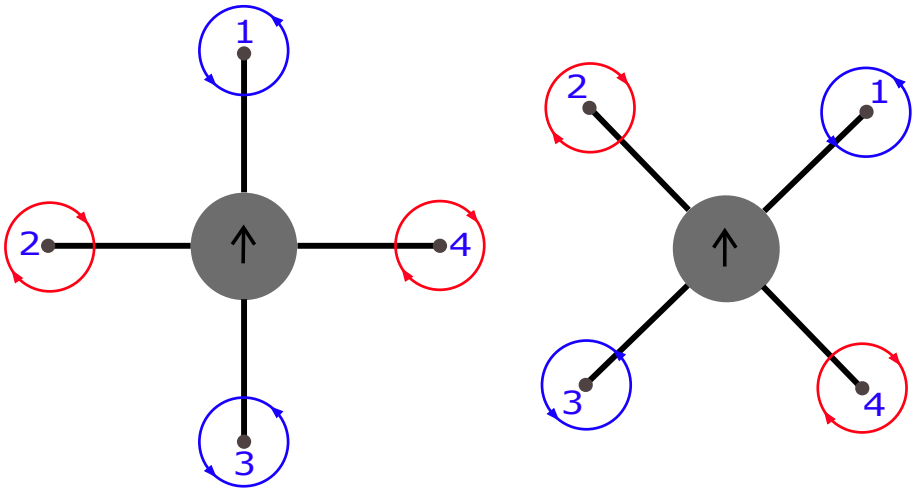


Obr. 1.1: Trikoptéra

formy je nestabilita počas letu a tiež odolnosť voči nárazovému vetru. Kvôli nízkemu množstvu motorov nie je vhodný na natáčanie videí a iné akcie, ktoré vyžadujú stabilitu. Trikoptéra nie je symetrická. Je potrebná prítomnosť servomotora pre otáčanie osamoteného motora, čím je kompenzovaná nerovnomernosť počtu motorov [3].

1.1.2 Kvadrokoptéry

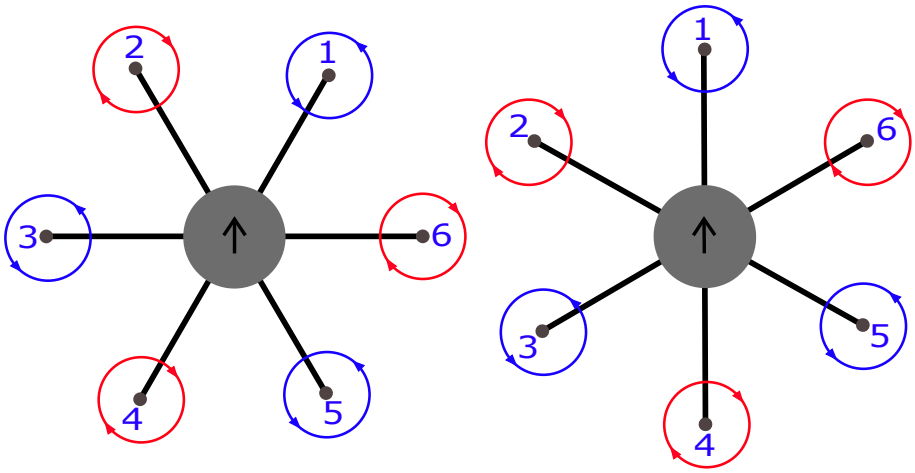
Majú štyri ramená a ku každému ramenu je pripojený jeden motor. Je vhodným kompromisom a poskytuje univerzálne využitie v interiéri aj v exteriéri. Z tohto dôvodu je tento druh platformy najvhodnejšou voľbou. Kvadrokoptéra je štvorvrtuľová a je najviac používaným druhom. Existujú dva typy prevedenia, a to typ X a typ +. Rozdiel je v smere letu voči motorom. Prvý, ktorý sa začal používať bol typ +. Počas pohybu UAV dopredu postačila zmena výkonu len u dvoch motorov. Častejšie používaný typ X využíva všetky motory pri každej zmene smeru letu. Hlavnou výhodou typu X je, väčší pozorovací uhol pre kameru. Hlavnou nevýhodou kvadrokoptéry je, že v prípade výpadku jedného motora ostatné nedokážu kompenzovať túto stratu [3].



Obr. 1.2: Kvadrokoptéra

1.1.3 Hexakoptéry

Šesť ramenné UAV so šiestimi motormi sa nazýva hexakoptéra a jej stabilita počas letu je podstatne vyššia. Hexakoptéry sú schopné udržať väčšiu hmotnosť a odolať nárazovému vetru. Taktiež existuje šanca, že pri výpadku jedného motora bude schopná hexakoptéra pristáť bez nejakého poškodenia. Nevýhodou je, že obratnosť takéhoto zariadenia je podstatne nižšia. Ďalšou nevýhodou je, že výsledná cena vzrastie z dôvodu potreby mať až šesť motorov. Pre ich väčší počet je taktiež po-

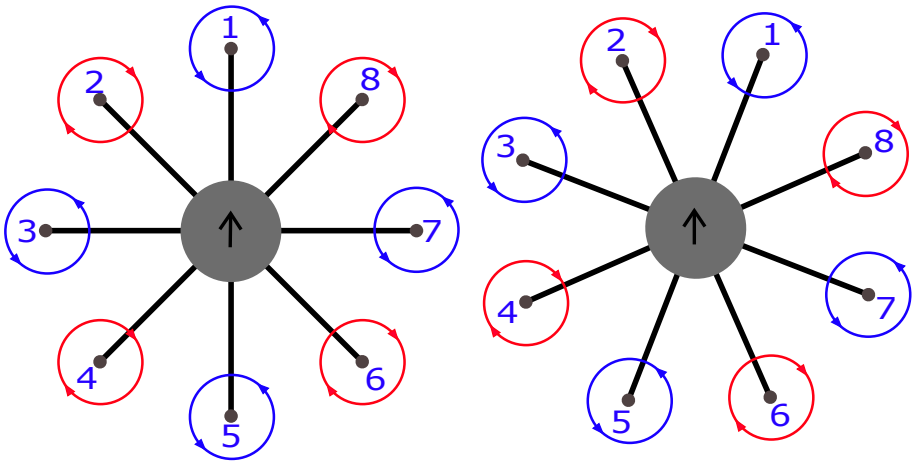


Obr. 1.3: Hexakoptéra

trebný väčší akumulátor, ktorý však zvýši hmotnosť celej hexakoptéry. Ako kvadrokoptéry, tak aj hexakoptéry môžu byť v dvoch usporiadaniach. V prvom prípade je let nasmerovaný medzi dve ramená, druhý spôsob spočíva v lete dopredu v smere jedného z ramien, viď obr. 1.3 [3].

1.1.4 Oktakoptéry

Oktakoptéry sú zo všetkých druhov najstabilnejšie. Ich veľkou výhodou je, v prípade výpadku jedného motora, že sú schopné stratu vykompenzovať a vo väčšine prípadov bezpečne pristáť. Oktakoptéry sú najvhodnejšie na prenášanie ťažkých predmetov, natáčanie stabilných záberov a sú vhodné pre skúsených pilotov. Ich nevýhodou je výrazná neobratnosť a vysoká cena [3].



Obr. 1.4: Oktakoptéra

1.2 Časti multikoptéry

Konštrukciu multikoptéry tvoria viaceré základné diely. Hlavným dielom je rám, na ktorý sa pripevnia všetky ostatné komponenty ako sú motory, vrtule, akumulátor, regulátory a riadiaca jednotka PixHawk.

Rám

Je hlavným nosným prvkom pre každý druh UAV. Najpoužívanejšie materiály pre tvorbu rámu sú karbón, zliatiny kovu alebo plast.

Karbón je najdrahšou možnosťou. Tento materiál je mimoriadne pevný a ľahký, ale jeho nevýhodu je krehkosť. Prevažne je využívaný práve pre závodné UAV.

V prevažnej väčšine sa na výrobu rámov používajú rôzne profily a rúrky zo zliatin kovov. Najčastejšie je to dural, ktorý je cenovo dostupný a ľahko získateľný. Z hľadiska hmotnosti sú tieto zliatiny ťažšie než karbón. Na druhú stranu, pre ich opracovanie nie je potrebné žiadne špeciálne náradie [3].

Pre výrobu rámu z plastu je najvhodnejšia metóda 3D tlače. Veľkou výhodou je možnosť navrhnúť si vlastný model a jednoducho si ho vytlačiť a poskladať. V prípade poškodenia niektorej časti je túto časť možné jednoducho nahradiť. Táto metóda je jednoduchá, rýchla a lacná, ale je potrebné mať prístup k 3D tlačiarni [3].

Motor

Motor je najnákladnejšou súčiastkou UAV. Magnet a cievka sú základnými komponentmi každého elektromotora. Prechodom prúdu cievkou sa roztočí motor, ktorý zároveň roztáča vrtuľu a tým dáva UAV potrebný vztlak k vzletu. Existujú dva druhy motorov, a to jednosmerné a striedavé. Jednosmerné motory sa využívajú hlavne pri malých modeloch. Častejšie využívaný striedavý motor je využívaný pri väčšine UAV. Parameter, ktorým sa odlišujú motory je KV. Znamená to počet otáčok za minútu na 1 Volt na prázdno. Motory s 500-1000 KV sú vhodné pre pomalé a stabilné lety. Motory 1100 KV a viac sa hodia pre akrobatické multikoptéry[5],[6].

Vrtule

Vrtule je možné vybrať z rôznych materiálov ako je plast, drevo a karbón. Najčastejšie používaným materiálom je plast. Výhodou menších rozmerov lopatiek je schopnosť rýchlej akcelerácie, preto sú vhodné pre akrobatické a závodné UAV. Naopak, veľké rozmery lopatiek sú vhodnejšie pre stabilný let. Vrtule sa označujú dvomi parametrami. Prvé číslo označuje priemer vrtule v palcoch a druhé stúpanie za jednu otáčku. Vhodná vrtuľa sa vyberá na základe parametrov, ktoré vyplývajú z vlastností

jednotlivých komponentov UAV. Tieto parametre sú vyjadrené v tabuľke každého motora [7].

Akumulátor

Akumulátor je súčiastka, ktorá uchováva a dodáva energiu celému UAV. Najdôležitejším parametrom akumulátora je kapacita v jednotke mAh. Kapacitu je nutné prispôsobiť rozmerom UAV a účelu, na ktorý bude využívaný, pretože akumulátor je najťažšou súčiastkou celého UAV. Na výdrž akumulátora nevplýva len hmotnosť celého UAV, ale aj štýl letu, teda jeho plynulosť. Najčastejšie používaný typ akumulátora je Li-po (Lithium Polymer). Existujú aj iné typy, ale tie sa používajú ojedinele. Li-Po má pri vysokej kapacite nízku hmotnosť a dokáže vydať veľkú energiu. Malou nevýhodou je, že je mierne drahší. Najčastejšie používané akumulátory sú typu 3 S alebo 4 S. Číslo označuje počet článkov a písmeno ich spôsob zapojenia. Písmeno S znamená zapojenie článkov v sérii a v prípade použitia paralelného zapojenia článkov sa používa písmeno P. Počet článkov v sérii udáva celkové napätie akumulátora. Parameter, ktorý udáva statický vybíjací prúd akumulátora je C. V prípade akumulátora 5 C je priemerná rýchlosť vybíjania batérie 5 násobok kapacity batérie [3],[6].

Regulátor

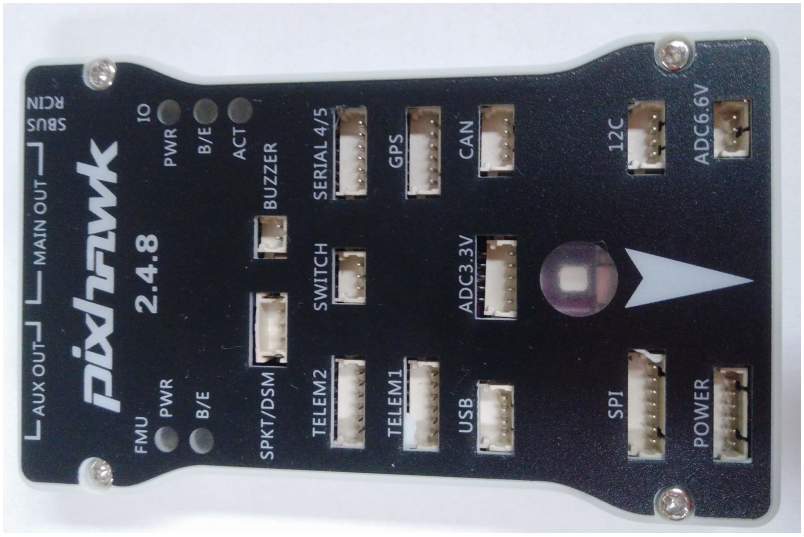
Elektronický regulátor otáčok je súčiastka, ktorá poskytuje možnosť riadiť energiu privádzanú do motora a tým zmenu jeho výkonu. Regulátor funguje na jednoduchom princípe privádzania napätia na jednotlivé póly pri rôznych frekvenciách. Hlavným parametrom je maximálny prúd, ktorý môže regulátorom pretekať. Jeho hodnota musí byť väčšia ako maximálny odoberaný prúd jedným motorom [6],[3].

PixHawk

Je nezávislý open hardware projekt, a to znamená, že informácie o hardvéri sú voľne dostupné zároveň so schémami zapojenia, zdrojovými kódmi, atď. Cieľom PixHawk je poskytnúť štandard pre rýchle, vysokokvalitné a nízkonákladové hardvérové návrhy pre priemyselné aj hobby komunity. Pixhawk podporuje ArduPilot a PX4. PX4 pracuje na operačnom systéme NuttX. NuttX je RTOS (Real-Time Operating System), čo v preklade znamená operačný systém, pracujúci v reálnom čase. Riadiaca jednotka letu Pixhawk má silný výpočtový výkon a je najpoužívanejším hardvérom s open source. Jeho predchodcom bol APM (ArduPilot Mega). Procesor APM však nebol dostatočne výkonný pre splnenie zložitejších procesov, preto bol prekonaný výkonnejším Pixhawk [8].

Pixhawk má v sebe zabudovaný gyroskop, akcelerometer a snímače tlaku. Jeho ovládače podporujú pripojenie externých zariadení, ako sú senzory a kamery.

V prípade natáčania videa, alebo inej činnosti, ktorá potrebuje výpočtový výkon, je potrebný samostatný počítač napríklad Raspberry Pi. Jednou z výhod používania rady Pixhawk sú flexibilita pripojenia periférií, dobrá kvalita zariadenia, veľká komunita a automatické aktualizácie firmware [9].



Obr. 1.5: Riadiaca jednota Pixhawk pre kontrolu UAV

ArduPilot je plnohodnotný a spoľahlivý open source software. Tento softvér sa rozvíja dlhšie ako 5 rokov. ArduPilot je jeden z mála auto pilotných softvérov, ktoré dokážu ovládať akýkoľvek typ vozidiel a to od bežných lietadiel, multikoptér a vrtuľníkov až po lode a ponorky. Pričom stále ďalej rozširuje pôsobenie tohto softvéru. ArduPilot nevyrába žiadny hardware, zato firmware funguje na širokom spektre hardvérových zariadení. Spolu so softvérom pozemného riadenia (GCS), sú zariadenia ako UAV s funkčným ArduPilot schopné komunikovať v reálnom čase s pilotom. Tiež môže prevádzkovať pokročilé nástroje pre zaznamenávanie, analýzu a simuláciu údajov [10].

Open source napomáha rýchlemu vývoju vďaka tomu, že mnoho prispievateľov vytvára rozhrania a následne ich zdieľa. Používatelia z toho profitujú tým, že využívajú existenciu portfólií, kde si môžu potrebné dáta nájsť, stiahnuť a upraviť podľa vlastných kritérií. Veľkou výhodou tejto platformy je existencia veľkej online komunity programátorov a používateľov, ktorá spolupracuje a je otvorená pomoci v prípade vzniku problémov [11].

1.3 Komponenty pre automatizovaný let

Na zaistenie automatizovaného letu je potrebný miniatúrny počítač, pripevnený na UAV, v ktorom bude fungovať softvér FlytOS. Pozemná riadiaca stanica je softvérom pre vzdialené ovládanie a nastavovanie UAV. Pre prvotné vytvorenie a následné otestovanie automatizovaného letu slúži simulátor FlytSIM, ktorý simuluje prostredie FlytOS.

FlytOS

FlytOS je softvérový framework od FlytBase. Framework je softvérová štruktúra, ktorá slúži ako podpora pri programovaní a vývoji. FlytOS je postavený na operačnom systéme ROS (Robot Operating System) a Linuxe. To z neho robí ideálnu platformu pre výskum škálovateľných aplikácií. Môže spolupracovať s akýmkoľvek kompatibilným UAV pomocou adaptérovej vrstvy(adapter layer), ktorá je sprístupnená pomocou FlytAPIs v rôznych jazykoch, ako sú napr C++, Python a Websocket [12].

FlytOS umožňuje pripojenie externých zariadení ako LiDAR (Light Detection And Ranging), kamier s vysokým rozlíšením, termokamier a pod. LiDAR je metóda diaľkového merania vzdialenosti. Ďalej existujú inteligentné moduly, ktoré poskytujú UAV schopnosti vyhýbať sa kolíziám, presné pristávanie na značke, sledovanie objektov, automatizovaný let, atď. Operačný systém FlytOS umožňuje rozhraniam API (Application Programming Interface) a súpravám SDK (Software Development Kit) vytvorenie aplikácií na vysokej úrovni, napríklad pre leteckú dopravu, poľnohospodárstvo, prieskumy a priemyselné kontroly. API je riešením pre jednoduchý rozvoj a má bohaté možnosti, ako napríklad vzdialené monitorovanie, kontrolu prostredníctvom webových a mobilných aplikácií [13],[14].

Operačný systém FlytOS je kompatibilný s väčšinou najpoužívanejších platforiem pre UAV a je podporovaný väčšinou populárnych kompaktných počítačov vrátane Nvidia, Odroid, Intel a Rasberry Pi.

Raspberry Pi

Je to miniatúrny počítač o veľkosti približne platobnej karty. Ide o takzvaný SoC (System on Chip). To znamená, že jediný čip obsahuje všetky komponenty počítača. Jeho jadrom je ARM procesor. Raspberry Pi neobsahuje displej a má len slot na MicroSD kartu. Ďalšie jeho periférie sú USB vstup, Ethernet, HDMI, Jack a GPIO(General Purpose Input and Output). Je možné naňho nainštalovať ľubovoľný operačný systém, ale musí byť kompatibilný s ARM platformou. V drvivej väčšine je to práve Linux v distribúcií Debian a Arch. Najrozšírenejším systémom je

Raspbian, založený na Debian. Najnovší model Raspberry Pi 3 už podporuje Wi-Fi a Bluetooth [15]. Raspberry Pi bol primárne vyvinutý pre účely výuky programovania, napríklad Java alebo Python. Často sa využíva pre mediálne zariadenia, ako napríklad domáce video centrum. Tiež zvláda prehrávanie videí v HD kvalite a má dekodér H.264/MPEG-4 AVC. Ďalej je ho možné využívať ako Web server a IP kameru. Vďaka pinom GPIO je možné k Raspberry Pi pripojiť veľa druhov zariadení ako napríklad senzory na meranie teploty alebo sledovanie vyťaženosti siete, a tým ich kontrolovať. Toto zariadenie má nespočetné množstvo spôsobov využitia [16].

Raspberry Pi model 3B má 64 bitový procesor 1,2 GHz, Wi-Fi 802.11n, bluetooth 4.1, gigabitový Ethernet a USB 2.0.

Ground Control Stations

Ground Control Stations v preklade pozemná riadiaca stanica je softvérová aplikácia. Používa sa na riadenie letu UAV a funguje zvyčajne na počítači alebo na nejakej prispôsobenej konzole. GCS má veľké množstvo schopností. Jednou z hlavých úloh riadiacej stanice je získavať a zobrazovať údaje, týkajúce sa letových parametrov bezpilotného lietadla. Tieto údaje sa môžu zobrazovať buď ako text alebo vizualizovať pomocou rôznych grafických prvkov. GCS komunikuje s UAV prostredníctvom bezdrôtovej komunikácie. Vďaka senzorom a zariadeniam na UAV nám môže GCS zobrazovať reálne dáta, ktoré sa odohrávajú na UAV. Napríklad aktuálna poloha, výkon motorov, smer letu, rýchlosť a tiež možnosť sledovania živého prenosu. GCS tiež slúži pre priamu kontrolu UAV. Prostredníctvom tejto aplikácie je možné nastavovanie a kontrolovanie letovej trasy UAV. Existuje veľa druhov aplikácií, ale používané a kompatibilné s FlytOS sú APM Planner a QGroundControl [17],[18],[19].

GCS je jedným z kľúčových komponentov pre UAS (Unmanned Aircraft System). UAS je systém a skladá sa z prvkov, ktoré sú nenahraditeľné pre let UAV. Zahrňuje v sebe UAV, komunikáciu, vybavenie, ako aj samotné GCS s pilotom.

QGroundControl je operátor riadiacej jednotky alebo inak povedané pozemný riadiaci softvér pre malé UAV. Jednou z jeho mnohých schopností je vizualizácia UAV v reálnom čase počas jeho prevádzky v interiér aj v exteriéri. Vďaka jeho flexibilnej softvérovej architektúre je schopný podporovať široký záber zariadení typu UAV. Jednou z hlavných postáv, ktorá prispela k tvorbe QGroundControl, bol Lorenz Meier. Jeho úlohou bolo pre projekt PIXHAWK vytvoriť softvér Ground Control Station (GCS). V súčasnosti QGroundControl dospel do štádia programu s otvoreným zdrojovým kódom. Tento softvér využíva veľká komunita s veľkým množstvom prispievateľov. Táto skutočnosť je výhodou a v prípade výskytu problému je veľká šanca nájdenia riešenia.

QGroundControl je open source a poskytuje plnú podporu pozemného riade-

nia letu a konfigurácie zariadenia. Podporuje veľa druhov bezpilotných zariadení prostredníctvom komunikácie MAVLink, vďaka ktorej je schopné riadiť zariadenia fungujúce na PX4 a ArduPilot. MAVLink (Micro Air Vehicle Link) je protokol, ktorý je používaný pre komunikáciu medzi GCS a UAV. QGroundControl má veľmi príjemné používateľské rozhranie, hlavne pre začiatočníkov. Napriek tomu poskytuje veľmi pokročilé funkcie, ako napr. na ovládanie letu a tiež aj pre samotnú konfiguráciu zariadenia. Tiež má ľahko ovládateľné plánovanie ciest pre autonómny let zariadenia. Pomocou mapy, ktorú si používateľ možné stiahnuť, je schopný bez prístupu k internetu sledovať aktuálnu polohu, smer a náklon UAV. QGroundControl je schopný zobrazovať trasu letu, trasové body a umožňuje zdieľanie živého prenosu z bezpilotného zariadenia [20].

QGroundControl dovoľuje vytvoriť len manuálne vložené body letu a tým navrhnúť trasu letu UAV. Inak povedané neposkytuje žiadny algoritmus plánovaného automatizovaného letu. Navyše umožňuje sledovanie misie iba jednému používateľovi v jednom momente. QGroundControl je jeden z mála GCS softvérov, ktoré tiež poskytujú možnosť kontroly viacerých UAV zariadení naraz. Jedna z jeho veľkých nevýhod je, že neobsahuje funkcionalitu NFZ (No-Fly Zones). NFZ slúži na zobrazenie priestorov, do ktorých je zakázaný let UAV a prípadnú možnú kalkuláciu tak, aby sa zariadenie tomuto priestoru vyholo [21].

FlytSIM

FlytSIM poskytuje simulačné prostredie 3D SITL (Software In The Loop). Softvér simuluje celkové prostredie FlytOS a podporuje základné aplikácie FlytAPI s možnosťou GCS a simulácie plánu letu a kontroly UAV. Taktiež slúži na testovanie aplikácií, ktoré zvyčajne vyžadujú hardvér pre overenie funkčnosti. 3D simulácia je založená na systéme ROS – Gazebo, ktorý simuluje prostredie, ale aj samotné UAV. Prostredie je generované podľa vopred stanovených parametrov a riadiace algoritmy, ktoré ovládajú UAV, sú rovnaké ako budú v skutočnom UAV. FlytSIM je primárne podporovaný na operačnom systéme Linux. Pomocou programu Docker je však možné jeho spustenie aj na Windowse 10 PRO a vyššie. Softvér FlytSIM je založený na Pixhawk SITL, ktorý je prispôsobený pre prácu s FlytAPI [22].

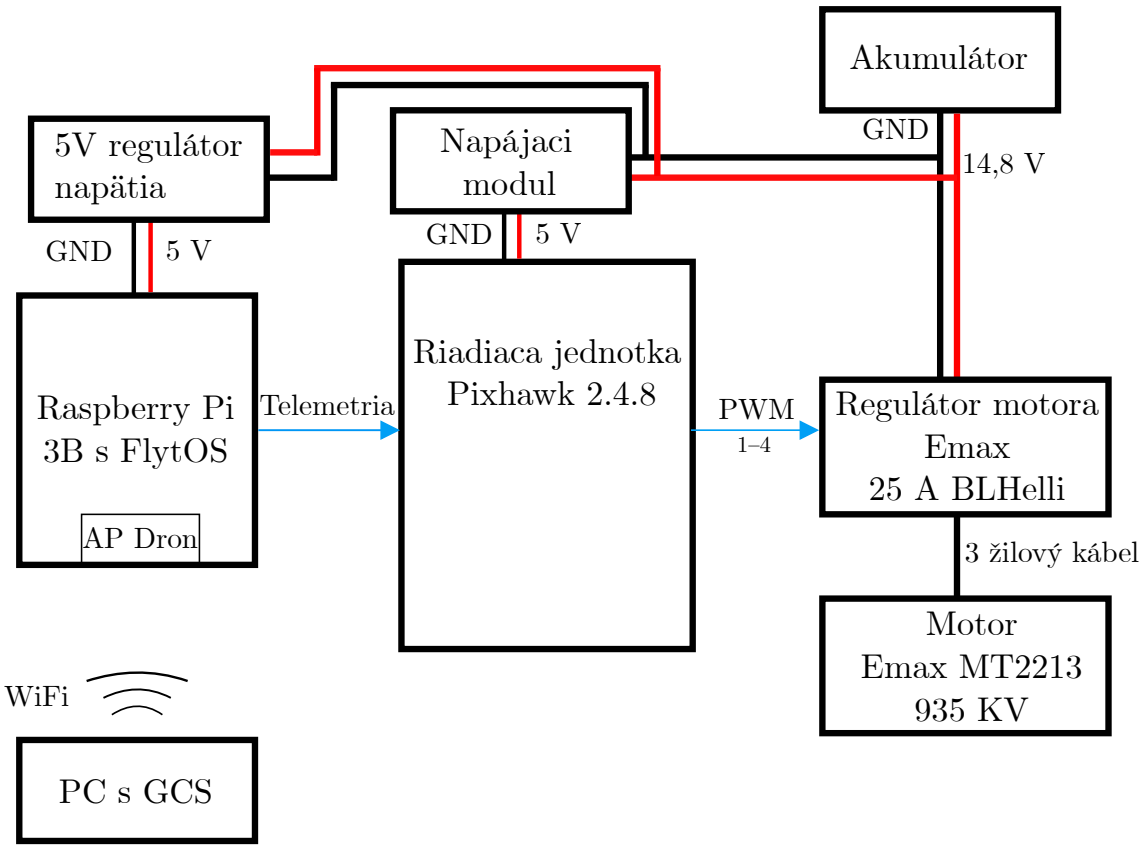
FlytSIM ma v základnom nastavení preddefinovaný APM a ten nie je podporovaný grafickým rozhraním Gazebo. Bez tohto rozhrania má nízke nároky na výpočtový výkon počítača a stále obsahuje všetky potrebné schopnosti pre vytvorenie simulácie. PX4 je schopný nielen simulovať, ale zároveň je tiež podporovaný grafickým rozhraním Gazebo [22].

2 Praktická časť

V praktickej časti je popísané zostrojenie kvadrokoptéry a výber vhodných súčiastok. Následne sa táto časť zameriava na inštaláciu FlytOS a zaistenie komunikácie medzi Raspberry Pi a Pixhawk. Praktická časť pokračuje zobrazením simulácie automatizovaného letu v simulátore FlytSIM a následne bol tento let otestovaný v reálnych podmienkach. Posledná časť sa zaoberá algoritmom pre vytvorenie automatizovaného letu, počas ktorého bude celý čas udržiavané smerovanie kvadrokoptéry na pozemnú stanicu.

2.1 Výber súčiastok

V tejto práci bola vybraná kvadrokoptéra. Z hľadiska stavby patrí k tým jednoduchším a finančne menej náročným. Z hľadiska letových vlastností patrí medzi univerzálne UAV, ktoré poskytujú dostatočnú pohyblivosť aj stabilitu. Hlavnou úlohou kvadrokoptéry je stabilný a pomalý let s možnou pridanou záťažou. Na základe týchto skutočností boli vyberané jednotlivé konštrukčné diely.



Obr. 2.1: Model zapojenia komponentov UAS

2.1.1 Rám

Ako najvhodnejšia metóda výroby rámu sa javí jeho 3D tlač. Táto metóda je najjednoduchšia a cenovo dostupná. V prípade pádu a poškodenia ktorejkoľvek časti rámu bude jednoduché vytlačiť náhradný identický kus. Z viacerých možných modelov bola vybraná predloha kvadrokoptéry DJI F450. Model bol stiahnutý zo stránky https://grabcad.com/library/dji-f450-quadcopter-drone-1. Má dostatočne vysoké nohy pre prípadnú inštaláciu prídavného zariadenia a sú od seba dostatočne vzdialené pre jednoduché vzlietanie a pristávanie. Ramená sú dosť dlhé na to, aby bola zabezpečená jednoduchá ovládateľnosť kvadrokoptéry. Na konci jednotlivých ramien sú už konštrukčne dopredu pripravené diery na uchytenie motora v štandarde $16 \times 19\,\text{mm}$. Jeho celková hmotnosť je $268\,\text{g}$. Kvadrokoptéra je vytlačená na 3D tlačiarni i3 MK3 od firmy Prusa. Táto tlačiareň je poskytnutá školou. Pre tlač dielov je použitý materiál PETG (Poly Ethylene Terephthalate Glycol) s $50\,\%$ výplňou. Model sa skladá zo 4 častí a to nohy, ramená, vrchná a spodná upínacia časť pre nohy a ramená.

Obr. 2.2: Diely rámu drona

Obr. 2.3: Výsledný tvar tela kvadrokoptéry

2.1.2 Motor

Od motora je očakávaná nosnosť približne jeden kilogram a jeho energetická nenáročnosť. Na splnenie zadaných kritérií bola vybraná konfigurácia s motorom EMAX 2216 - 810 KV. Tento motor je nízkootáčkový a ideálny pre stabilný let. Pri praktickej realizácii však nebolo možné tento typ motora obstarať v obchode a preto bola vybraná iná alternatíva a to nízkootáčkový motor EMAX MT2213 KV935, viď obr. 2.4. Motor je drahší, ale je dostupný z obchodu priamo v Brne. Očakávaná

Obr. 2.4: Motor EMAX MT2213 KV935

nosnosť jedného motora je 1 kilogram a to znamená, že celá kvadrokoptéra by mala uniesť 4 kg. Ďalej je potrebné, aby 2 motory boli CW, čo znamená, že sa budú otáčať v smere hodinových ručičiek a 2 motory CCW sú tie, ktoré sa otáčajú v protismere hodinových ručičiek. Tieto motory majú uchytenie v štandarde 16×19 mm. Toto uchytenie je vyhovujúce pre náš model UAV, ktorý má presne takéto umiestnenie dier na ramene.

Tab. 2.1: Parametre motora EMAX 2216 - 810 KV

Napätie	**Veľkosť vrtúľ**	**Prúd**	**Ťah**	**Výkon**	**Výkonnosť**	**Rýchlosť**
[V]	[-]	[A]	[g]	[W]	[g/W]	[RPM]
14,8	EMAX8045	1	130	14,8	8,8	3900
		2	230	29,6	7,8	5180
		3	310	44,4	7,0	6000
		4	390	59,2	6,6	6610
		5	470	74,0	6,4	7200
		6	530	88,8	6,0	7570
		7	580	103,6	5,6	7910
		8	630	118,4	5,3	8230
		9	670	134,2	5,0	8500
		10	700	148,0	4,7	8780
		10,7	720	158,4	4,5	9030

2.1.3 Vrtule

Pre nami vybraný motor sú navrhované vrtule s priemerom 8 palcov a stúpaním 4,5 palca, alebo priemerom 10 palcov a stúpaním 4,5 palca. Je potrebné zabezpečiť, aby kvadrokoptéra dokázala lietať nielen sama, ale aj s potenciálnym pridaným závažím. Z tohoto dôvodu boli vybrané plastové vrtule 1045, ktorých výhodou je, že už sú súčasťou balenia motorov, viď obr. 2.5.

Obr. 2.5: Plastové vrtule 1045

2.1.4 Akumulátor

Zvolený akumulátor od Gens Ace je 4S1P má kapacitu 4000 mAh, viď obr. 2.6. Je typu Li-Po a jeho trvalý vybíjací prúd je 25 C, čo znamená, že je schopný trvalo dodávať 100 A pri napätí 14,8 V. Akumulátor má hmotnosť 438 g a maximálny nabíjací prúd je 1 C, čo je 4 A.

Obr. 2.6: Akumulátor Gens Ace 4000 mAh

2.1.5 Regulátor

Regulátory sú vyberané podľa maximálneho prúdu, ktorý preteká motorom. Táto hodnota by nemala presiahnuť 20 A. K maximálnemu prúdu sa musí pripočítať určitá rezerva, preto by mal byť vhodný regulátor s hodnotou 25 A. V danom rozsahu nebol žiadny na sklade, preto bol vybraný regulátor 30 A od GPX Extreme, viď obr. 2.7. Tieto regulátory môžu trvalo dodávať prúd 30 A a zvládnu nárazový prúd až 40 A po dobu 10 sekúnd. Sú vhodné pre 2 až 4 článkové akumulátory.

Obr. 2.7: Regulátor Emax BLHeli 25 A

2.2 Inštalácia FlytOS

FlytBase je prvá svetová IoD (Internet of Drones) platforma. Stránka FlytBase poskytuje bezplatné vytvorenie účtu a pripojenie 5 zariadení na osobné účely. Zo stránky https://flytbase.com/ bol stiahnutý potrebný software. Nasleduje vybranie

zariadenia zo zoznamu, ktoré je kompatibilné s FlytOS. V tejto práci je použité zariadenie Raspberry Pi model 3B. Ďalším krokom je inštalácia operačného systému. Je možné využiť inštaláciu Linuxu s implementovaným FlytOS, alebo druhou možnosťou je inštalácia Linux - Ubuntu 16.04 s následnou inštaláciou FlytOS.

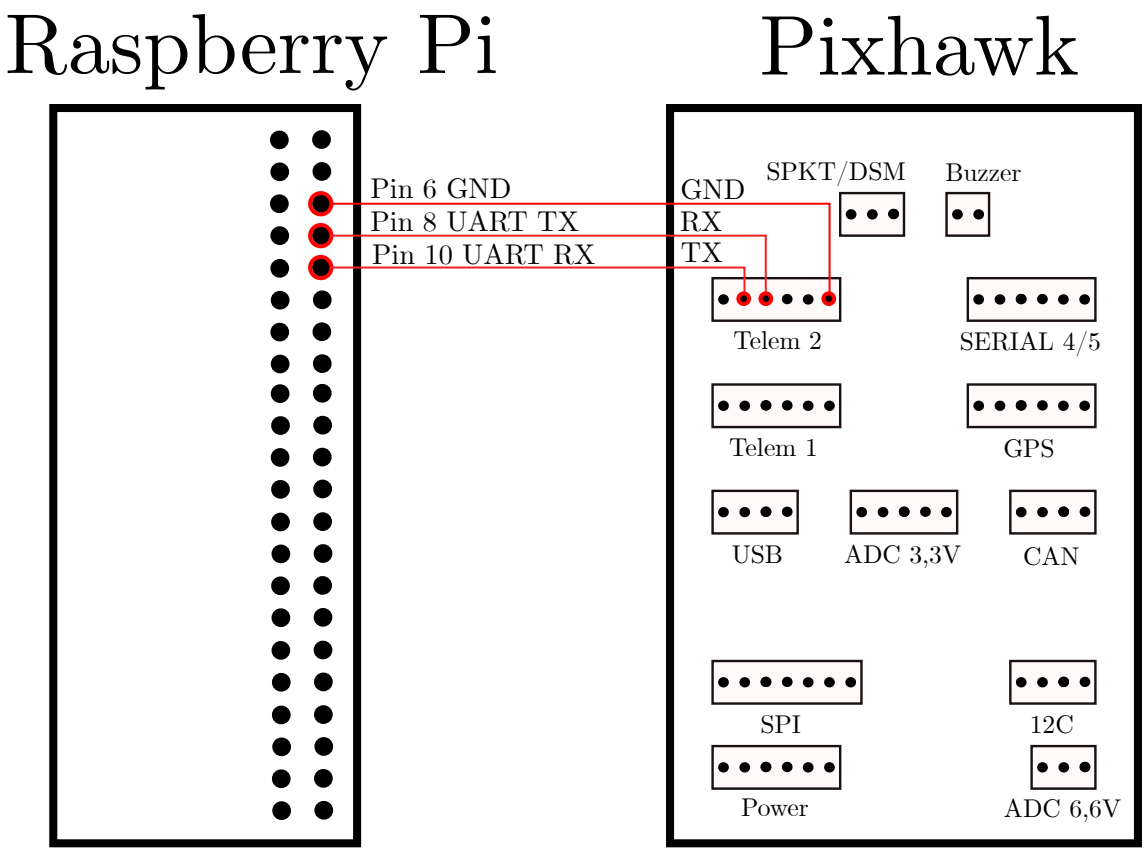
Pri využití možnosti stiahnutia operačného systému Linux s už implementovaným FlytOS, systém po spustení na Raspberry Pi detekoval chybu a následne nebolo možné zariadenie opäť spustiť.

Po reinštalácií bolo možné zapnúť konzolu FlytOS, ale v krátkych a nepravidelných intervaloch začal samotný systém mrznúť. Z tohto dôvodu bola vyskúšaná druhá metóda inštalácie samostatného Linuxu a následná inštalácia FlytOS. Tento variant však tiež začal po nejakom čase zamŕzať a príčinu tejto chyby nebolo možné zistiť. Prvým pokusom o nápravu bolo vyskúšanie zmeny SD karty z dôvodu overenia jej funkčnosti. Táto zmena túto závadu nevyriešila a preto bolo potrebné vymeniť samotné Raspberry Pi 3B. Na novom zariadení je nainštalované FlytOS pomocou prvej metódy inštalácie a funguje bez problémov. Ďalším krokom bolo otvorenie konzoly FlytOS a overenie zariadenia pre uznanie licenčných podmienok. Po aktivácii bol FlytOS plne funkčný. Tým bola inštalácia ukončená.

V ďalšej časti bolo potrebné aktualizovať firmware riadiacej jednotky Pixhawk. Aktualizácia prebiehala po pripojení USB k počítaču, na ktorom bol spustený program QgroundControl. V tomto programe boli preddefinované parametre aktualizácie firmwaru a umožnený výber z dvoch druhov firmwaru a to PX4 a APM. Z týchto možností bol vybraný Ardupilot(APM), ktorého výhodou je väčšia komunita používateľov a je viac využívaný.

Dôležitým krokom je prepojenie Pixhawk a Raspberry Pi tak, aby mohli vzájomne komunikovať. Toto prepojenie je zaistené vlastnoručne vyrobeným spojom, viď obr. 2.8. Riadiaca jednotka je napájaná pomocou napájacieho modulu. FlytOS poskytuje vlastné grafické rozhranie (GCS) pre konfiguráciu riadiacej jednotky, nastavovania telemetrie, kalibráciu senzorov a sledovanie aktuálneho diania na UAV. Niektoré z uvedených schopností sú podporované len PX4. APM pre využitie plného potenciálu potrebuje externú kontrolnú riadiacu stanicu. Vybraná GCS QgraundControl je plne kompatibilná s APM a je voľne dostupná.

Overenie prepojenia a konfigurácie nastavení prebiehala nasledovne. Bol vytvorený AP (Access point) na Raspberry Pi s pripojenou telemetriou k Pixhawk. Následne bolo spustené GCS na počítači, ktorý bol pripojený k AP Raspberry Pi. Po vykonaní týchto krokov bolo možné sledovanie aktuálneho diania na Pixhawk, ktorý má implementovaný gyroskop a pripojený GPS modul, vďaka ktorému bolo možné zobrazenie jeho aktuálnej polohy.



Obr. 2.8: Prepojenie Telemetrie Pixhawk a Raspberry Pi

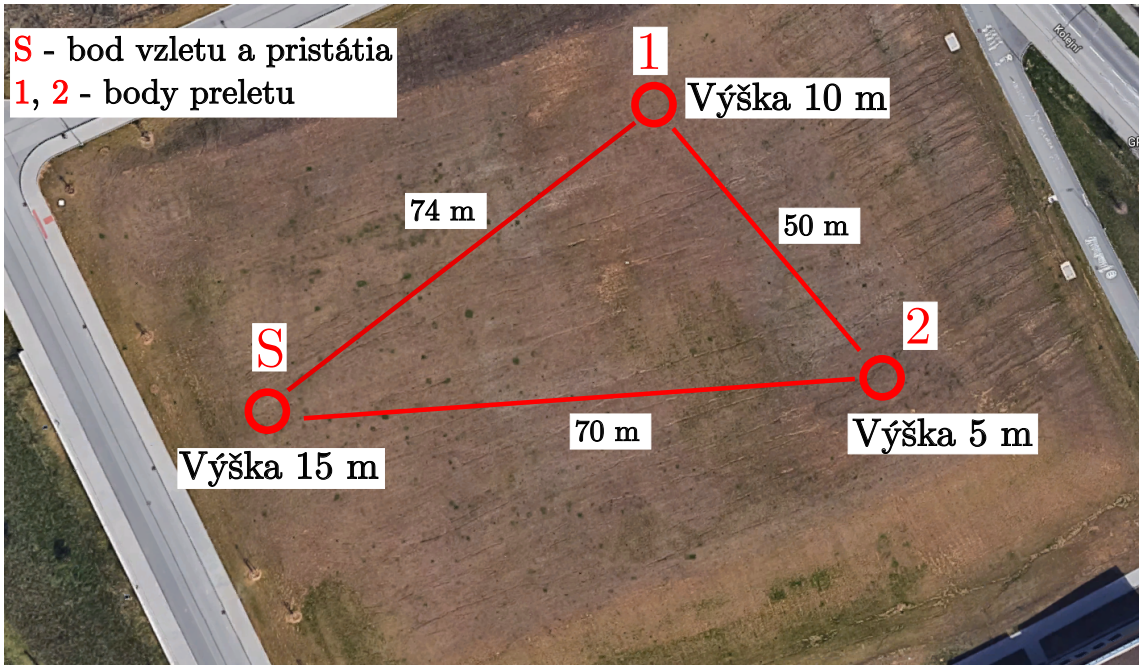
2.3 Simulácia letu vo FlytSIM

Hlavnou úlohou simulácie je vytvorenie a hlavne otestovanie, že automatizovaný let bude v reálnych podmienkach fungovať. Napísaný kód pre ovládanie kvadrokoptéry musí byť možné nahrať do Raspberry Pi. Po zložení tela kvadrokoptéry a pripojení všetkých potrebných dielov sa očakáva rovnaké správanie ako pri simulácii. Vo FlytSIM bola vytvorená simulácia na voľnom priestranstve vedľa školy. Plán letu je nastavený tak, že kvadrokoptéra vzlietne, preletí dvomi bodmi, natočí sa na miesto z ktorého vzlietla a vráti sa na miesto z ktorého vzlietla, viď obr. 2.9. V budúcnosti by mala kvadrokoptéra poskytovať prístupový bod pre určitú oblasť. Pritom musí nepretržite udržiavať vysokorýchlostné spojenie so základovou stanicou. Z toho dôvodu musí byť anténa na kvadrokoptére natočená na jej polohu počas celého letu.

Vo FlytSIM je možné navrhnúť automatizovaný let viacerými spôsobmi. Najjednoduchšia cesta je prostredníctvom grafického rozhrania. Nastavia sa body na mape, ktorými chceme aby kvadrokoptéra preletela a potvrdia sa. Po odoslaní príkazu na vykonanie kvadrokoptéra vykoná príkaz. Táto metóda je v budúcnosti nevyužiteľná, pretože kvadrokoptéra je vždy natočené v smeru letu.

Ďalšou metódou je nastavovanie smeru letu na osiach x, y a z. V tejto metóde

je už možné nastavovať natočenie kvadrokoptéry počas letu. Táto možnosť ovládania je však veľmi nepraktická, pretože kvadrokoptéra za reálnych podmienok musí vzlietnuť presne z toho istého miesta ako v simulácii. Poslednou nevýhodou je, že kvadrokoptéra nie je schopná lietať mimo smeru ôs.

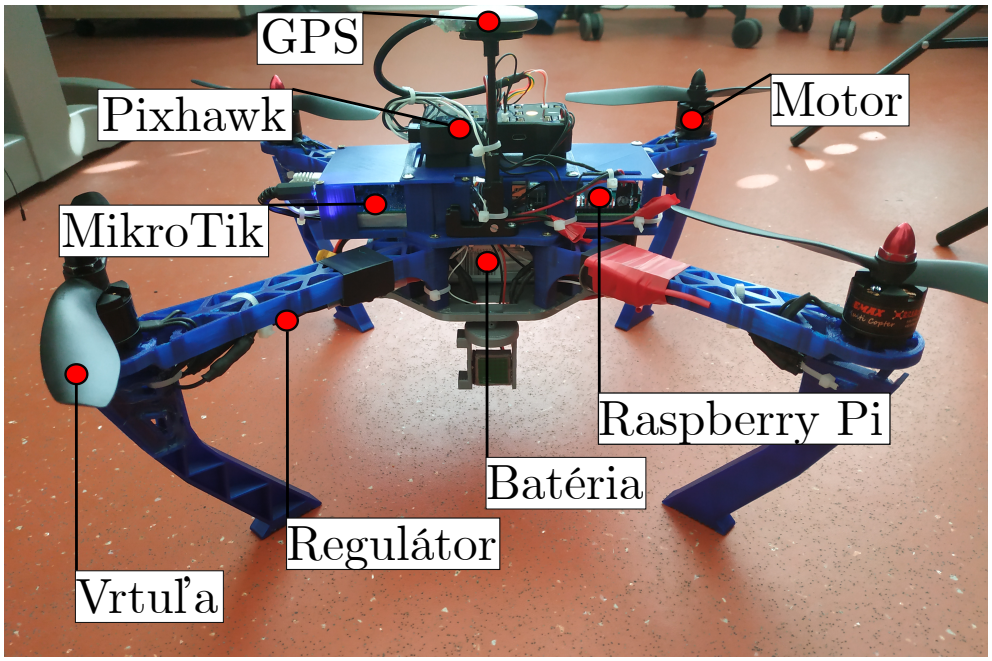


Obr. 2.9: Trasa automatizovaného letu z bodu S cez bod 1 a 2 a pristátie na bode S

Najvhodnejšou metódou pre simuláciu je nastavovanie bodov letu pomocou GPS súradníc. Táto metóda nám dovoľuje nastavovať akúkoľvek polohu na zemi. Tiež poskytuje možnosť zmeny výšky a nastavovanie natočenia kvadrokoptéry počas letu. Pomocou tejto metódy bola vytvorená simulácia automatizovaného letu viď prílohu A. Kvadrokoptéra začína na bode S a následne sa vznesie do výšky 5 metrov. Pokračuje na bod 1, počas letu vystúpi na výšku 10 m a natočí sa prednou stranou na bod S. Následne letí na bod 2, klesne na výšku 5 m a natočenie kvadrokoptéry sa nemení. V bode 2 zmení smer letu na bod S, kde počas letu vystúpa na výšku 15 m. Po dosiahnutí bodu S pristane a motory sa zastavia, viď obr. 2.9.

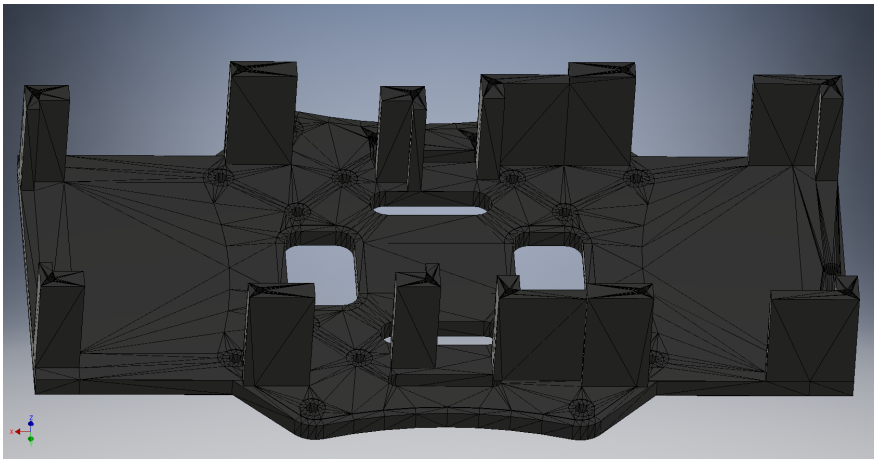
2.4 Zostrojenie kvadrokoptéry

Na obr. 2.10 je zobrazene UAV typu DJI F450 s novým druhom nôh, ktoré sú vhodnejšie pre bezpečné vzlietanie a pristávanie. Na telo kvadrokoptéry DJI F450 boli nainštalované všetky komponenty potrebné pre let a to motory, vrtule, radiče, riadiaca doska, modul pre batériu, batéria a riadiaca jednotka.



Obr. 2.10: Zostrojená kvadrokoptéra DJI F450 s pripevneným Raspberry Pi a Mikrotik

Pre automatizovaný let je potrebný GPS modul a Raspberry Pi na ktorom beží FlytOS. Pre nedostatok miesta na kvadrokoptére je Raspberry Pi napájané od riadiacej jednotky Pixhawk prostredníctvom GPIO pinov. Pre pripevnenie Raspberry Pi a Mikrotik na kvadrokoptéru bolo potrebné vytlačiť prídavné púzdro, do ktorého boli tieto zariadenia vložené viď obr. 2.11. Na úplnom vrchu kvadrokoptéry je umiestnená riadiaca jednotka Pixhawk, ku ktorej sú následne všetky komponenty pripojené.



Obr. 2.11: Púzdro pre upevnenie Raspberry Pi a Mikrotik na kvadrokoptéru

Po zostrojení tela kvadrokoptéry bola vykonaná kalibrácia senzorov riadiacej jednotky a GPS modulu tak, aby UAV rozpoznalo príkazy, ktorým smerom má

letieť. Dôležitým krokom bola konfigurácia komunikačných kanálov pre kontrolu UAV a pre zmenu módov lietania. Prvé 4 kanály sú nastavené pre kontrolu smeru a rýchlosti letu UAV. Do ostatných kanálov sa nastavujú módy letu a jedným z nich musí byť GUIDED MODE. Tento mód slúži na prepnutie riadenia z diaľkového ovládača na FlytOS, ktorý má od toho okamžiku plnú kontrolu nad UAV. Všetky konfigurácie boli vykonávané po pripojení riadiacej jednotky k počítaču, na ktorom bol spustený program Mission Planner.

2.5 Testovací let

Počas testovacích letov boli otestované všetky plánované možnosti ovládania kvadrokoptéry. UAV bolo ovládane prostredníctvom diaľkového ovládača, mobilnej aplikácie a prostredníctvom PC.

2.5.1 Manuálny let

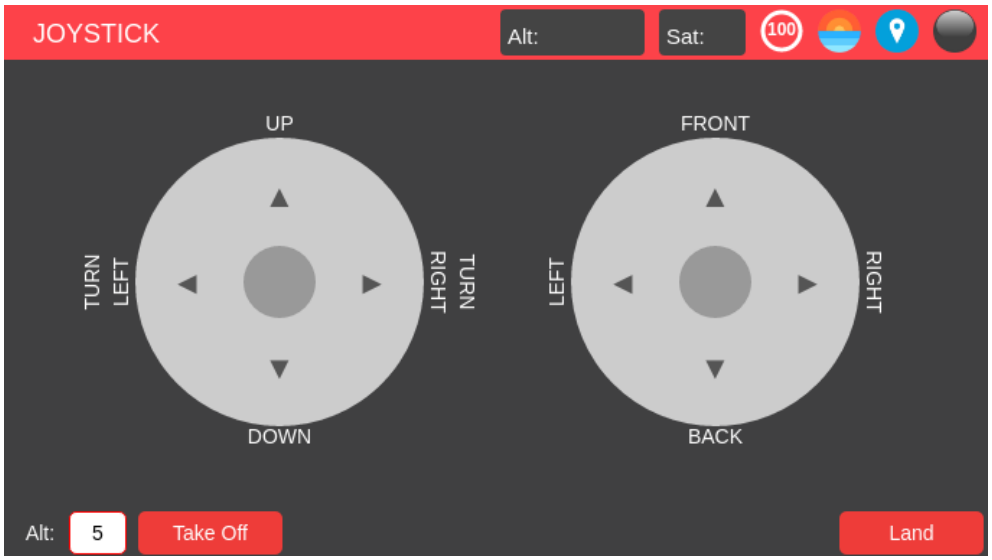
Po dobu prvého letu bola kvadrokoptéra ovládaná len prostredníctvom diaľkového ovládania tak, aby boli otestované všetky jej letové schopnosti a overená jej funkčnosť. Let bol úspešný a jej letecké schopnosti, tak ako aj výdrž batérie dopadla podľa očakávaní. V pokojnom tempe na polovicu nabitú batériu dokázala letieť necelých 8 minút.

Počas ďalšieho letu bolo ovládanie funkčné, ale po krátkom čase kvadrkoptéra havarovala. Väčšina rámu bola zničená. Nakoľko telo kvadrokoptéry bolo vytlačené na 3D tlačiarni, opakovaná tlač umožnila o necelý týždeň vykonávať ďalšie testy. Po dlhšom hľadaní chyby bolo zistené, že jedna z vrtúľ nebola dostatočne pripevnená k motoru a to spôsobilo pád kvadrokoptéry.

Pri ďalšom pokusnom lete sa kvadrokoptéra pri vznášaní nečakane zrútila. Po dlhšom hľadaní chyby sa zistilo, že riadiaca jednotka Pixhawk bola preťažená, pretože napájala aj Raspberry Pi a tým bola zničená. Následne bola riadiaca jednotka vymenená za novú a pre Raspberry Pi bol pripevnený vlastný napájací modul.

Následne bolo na kvadrokoptéru pripevnené Raspberry Pi a testované ovládanie kvadrokoptéry prostredníctvom FlytOS cez program Mission Planner na počítači, ktorý bol pripojený do siete Raspberry Pi. Testované bolo konkrétne sledovanie aktuálneho diania na UAV, roztočenie jeho motorov či vzlietnutie do určitej výšky a sledovanie aktuálnej polohy. V prípade, ak by sa vyskytla neočakávaná situácia, bolo možné prepnúť komunikačný kanál kvadrokoptéry späť na diaľkove ovládanie a tým ho dostať pod kontrolu.

Pri poslednom manuálnom teste bola vyskúšaná mobilná aplikácia pre kontrolu UAV. Pred spustením aplikácie je potrebné byť pripojený do siete Raspberry Pi. Po



Obr. 2.12: Mobilná aplikácia pre kontrolu UAV

spustení sa nastavení IP adresa Raspberry Pi a zobrazí sa grafické prostredie pre kontrolu UAV. Na obr. 2.12 je táto aplikácia zobrazená a je s ňou možné kontrolovať UAV, ako napríklad vzlet, pristátie, zmenu výšky a smer letu. APM nepodporuje možnosť otáčania UAV do ľava a do prava. Mobilná aplikácia tiež zobrazuje aktuálnu výšku a počet viditeľných satelitov. Veľkou nevýhodou je dlhá odozva kontroly UAV preto bolo vyskúšané len vzlietnutie a pristátie. Aplikácia je voľne dostupná a je možné si ju bezplatne stiahnuť na stránkach FlytOS.

2.5.2 Automatizovaný let

Dôležitým krokom testovania FlytOS bol prvý automatizovaný let kvadrokoptéry a to test základnej demoaplikacie priamo vo FlytOS. Spustenie prebehlo prostredníctvom vzdialeného pripojenia k Raspberry Pi cez PuTTY na počítači. Tento automatizovaný let prebiehal tak, že UAV vzlietne, preletí tvar štvorca o rozmere 5 metrov a pristane. Tento test prebehol úspešne.

Druhý automatizovaný let bol simulovaný vo FlytSIM a prebiehal podľa obr. 2.9. Tento test bol kľúčový, nakoľko sa od neho očakávalo konkrétne natáčanie kvadrokoptéry. Výsledky dopadli podľa očakávania a preto bolo možné pokračovať a navrhnúť algoritmus, ktorý by pri zadaní akejkoľvek súradnice vypočítal natočenie UAV na základovú stanicu tak, aby vždy mohlo byť medzi anténami udržiavané vysoko rýchlostné spojenie.

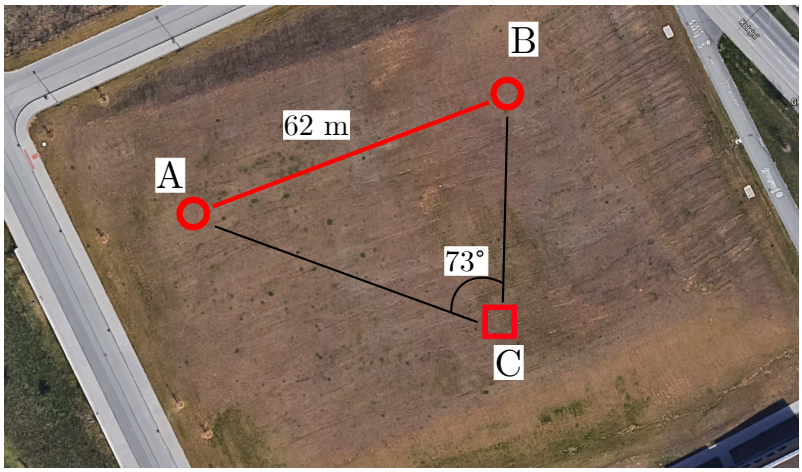
Obr. 2.13: Automatizovaný let kvadrokoptéry zo simulácie

FlytOS umožňuje nastavenie rýchlosti letu ale tento parameter nefungoval. Neskôr bolo zistené že nastavenie rýchlosti je podporované len PX4 (firmvér riadiacej jednotky) a v tejto kvadrokoptére bol využívaný APM. Tento nedostatok je jedným z chýb APM pri využívaní FlytOS. Ďalším nedostatkom je potreba externej GCS pričom PX4 má možnosť všetko konfigurovať vo vlastnom grafickom rozhraní vo FlytOS. Následne PX4 podporuje využitie grafického zobrazenia simulácie vo FlytSIM. Pre tieto dôvody je doporučené používať PX4.

2.6 Algoritmus pre výpočet súradníc a ich natočenie

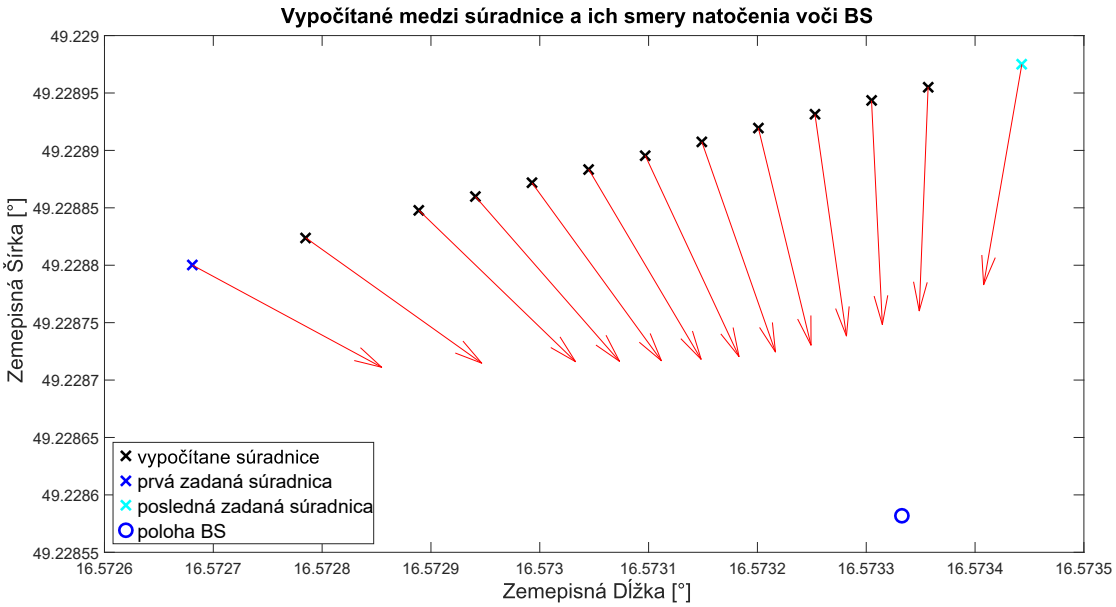
Dôležitou časťou zadania je vytvorenie skriptu, do ktorého budú zadané súradnice základovej stanice, všetky ďalšie súradnice letu, ich výška a doba čakania v určených bodoch. Výstupom skriptu bude Python kód, ktorý bude možné spustiť vo FlytOS.

Na obr. 2.14 je znázornená trasa letu plánovanej simulácie. Z dôvodu udržania vysoko rýchlostného spojenia počas preletu medzi bodom A a B musí UAV celý čas udržiavať natočenie na základovú stanicu, teda na bod C. Anténa na UAV má svoj maximálny vyžarovací uhol a počas letu sa nesmie stať, aby prestala so základovou stanicou komunikovať. V nasledujúcom príklade je nastavená odchýlka 10 °, a to znamená, že medzi dvomi bodmi letu nesmie byť väčší uhol ako 10 °. Na obr. 2.14 je znázornený uhol medzi dvomi bodmi letu a to 73 °, preto je potrebný výpočet medzibodov letu viď obr. 2.15. V budúcom využívaní sa plánuje použiť MikroTik wAP 60 GHz.



Obr. 2.14: Plánovaný automatizovaný let z bodu A do bodu B, bod C je základová stanica

Obrázok 2.15 obsahuje prvú a druhú zadanú súradnicu a všetky vypočítané medzisúradnice. Krok vypočítaných medzisúradníc sa mení podľa toho, či splní podmienku rozdielu maximálnej odchýlky medzi súradnicami a ak nie, tak sa krok skráti.



Obr. 2.15: Zobrazený výstup skriptu kde každá súradnica v bode letu má nastavený smer na základovú stanicu

Algoritmus 1 pre výpočet medzibodov letu a ich natočenia prebieha nasledovne. Na začiatku algoritmu sú vyžadované vstupné údaje ako počet bodov letu, súradnice základovej stanice a maximálna dovolená odchýlka medzi bodmi letu. V prvom

cykluse while sú vložené súradnice všetkých bodov letu. Algoritmus pokračuje ďalším cyklusom while kde je overené, či súradnice nie sú na rovnakej zemepisnej šírke alebo dĺžke ako základová stanica. Ak nie sú, tak sú vypočítané uhly natočenia prvej a druhej zadanej súradnice a následne je medzi nimi vypočítaný rozdiel. Ak je tento rozdiel menší ako zadaná maximálna odchýlka, uloží sa vypočítané natočenie a súradnice letu do výstupných polí. V prípade ak je rozdiel väčší ako zadaná odchýlka, tak sa rozdelí prelet na viac bodov letu podľa vypočítaného kroku a v každom kroku je vypočítaný uhol natočenia UAV. Každý krok je overovaný či v ňom platí podmienka maximálnej odchýlky. V prípade ak je táto odchýlka stále väčšia, tak je krok zmenšený. Všetky hodnoty sú uložené do polí a na konci algoritmu sú uložené do výstupného súboru tak, aby ho bolo okamžite možné použiť vo FlytOS viď prílohu C.

Boli vyskúšané dva tvary uloženia pre FlytOS. Prvý tvar pre vytvorenie misie disponoval všetkými možnosťami ktoré sme vyžadovali ale hlavný parameter otáčania počas letu nebol funkčný a nebolo zistené prečo. Z tohoto dôvodu bol vybraný formát GPS súradníc ako v prvej simulácií, ktorá nemá toľko možností nastavení automatizovaného letu ale je postačujúca a predovšetkým funguje parameter natočenia. Výstup tohoto algoritmu viď prílohu B.

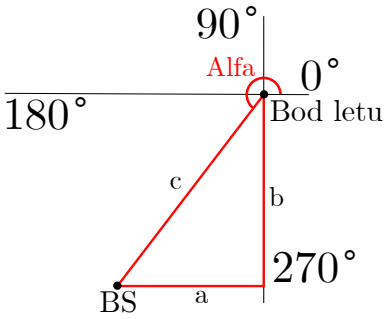
Algoritmus 1: Zobrazený výstup skriptu

```
1  input("Zadaný počet bodov preletu");
2  input("Zadaná zemepisná šírka a dĺžku BS: ");
3  input("Medzi odchýlka v stupňoch medzi bodmi letu: ");
4  while opakuj podla toho koľko je bodov preletu do
5      zadaj zemepisnú šírku a dĺžku budu;
6      zadaj výsku letu v bode a dobu čakania;
7  while
8      opakuj podla toho koľko je bodov preletu do
9      Vypočítaj a ulož do výstupu cez funkciu výpočet uhol prvej súradnice voči
        BS;
10     if súradnice bodu != súradnicami BS then
11         vypočítaj uhli pre prvú a druhu súradnicu letu voči BS;
12     else
13         print('bod letu je na rovnakej súradnici ako BS');
14     alfa = vypočítaj rozdiel uhlu medzi prvým a druhým bodom;
```

```
(1)
(2)  if alfa > zadaná citlivosť then
(3)      vypočítaj krok zemepisnej šírky a dĺžky;
(4)      pripočítaj krok k prvej súradnici zemepisnej šírky a dĺžky;
(5)      f = 1;
(6)      while f == 1 do
(7)          puhol[1] = vypočítaj uhol novej súradnice s krokom voči BS;
(8)          puhol[2] = vypočítaj rozdiel uhlov medzi prvou súradnicou a prvou s
              krokom;
(9)          while puhol[2] > nastavená citlivosť do
(10)             zmenši krok šírky a dĺžky o polovicu pri každom opakovaní;
(11)             puhol[1] = vypočítaj uhol novej súradnice so skráteným krokom;
(12)             puhol[2] = vypočítaj novy rozdiel uhlov medzi prvou súradnicou
                  a prvou so skráteným krokom;
(13)         cislo = rozdiel medzi prvou súradnicou a prvou s krokom;
(14)         if cislo < rozdiel uhlu medzi prvou a druhou zadanou súradnicou
              then
(15)             ulož do výstupnú uhol, zemepisnú šírku a dĺžku s krokom;
(16)             ulož výsku letu z prvej strednice pre všetky medzi súradnice
                  medzi prvou a druhou súradnicou;
(17)             ulož dobu čakania 0 pre medzi súradnice;
(18)             sirka = sirka + krok zemepisnej šírky;
(19)             dlzka = dlzka + krok zemepisnej dĺžky;
(20)         else
(21)             f = 0;
(22)     ulož do výstupu poslednú výsku, dobu čakania, zemepisnú šírku, dĺžku z
          terminálu a uhol natočenia;
(23)     while opakuj podľa dĺžky poľa súradníc do
(24)         funkcia vypocet2 vypočítaj uhol natočenia UAV na BS a ulož;
(25)     Ulož všetky vypočítane údaje do výstupného súboru let.py v tvare pre
          FlytOS;
```

2.6.1 Výpočet uhlu otáčania UAV

Do algoritmu 2 sú vložené údaje o zemepisnej šírke, dĺžke základovej stanice a bodu letu. Uhol natočenia kvadrokoptéry je vypočítaný cez funkciu tangens a strany trojuholníka sú získané z vypočítaného rozdielu zemepisných šírok a dĺžok. Tento rozdiel je vynásobený vzdialenosťou 1 ° na metre. Výsledný uhol je vypočítaný v stupňoch a uhol zodpovedá uhlom v jednotkovej kružnici. Stred jednotkovej kružnice je počítaná súradnica viď obr. 2.16.



Obr. 2.16: Zobrazený výpočet natočenia v bode letu voči základovej stanice BS

Algoritmus 2: Funkcia výpočtu uhlu natočenia UAV voči zákldaovej stanici

```
(1)  def vypocet2(bs,bd,ys,yd): # zemepisná šírka, dĺžka BS a bodu letu
(2)  a = (yd - bd) * d # výpočet dĺžky v metroch;
(3)  b = (ys - bs) * s # výpočet šírky v metroch;
(4)  alfa = degrees(atan(a / b)) # výpočet uhlu;
(5)  alfa = sqrt(alfa ** 2);
(6)  if ak sa súradnica nachádza voči BS v 1 kvadrante then
(7)      alfa = 270 - alfa;
(8)  else if ak sa nachádza v 2 kvadrante then
(9)      alfa = 270 + alfa;
(10) else if ak sa nachádza v 3 kvadrante then
(11)     alfa = 90 - alfa;
(12) else if ak sa nachádza v 4 kvadrante then
(13)     alfa = alfa + 90;
(14) else
(15)     print('chyba pri vypocte trojuholnika a urcovani kvadrantu');
(16) return (alfa,a,b)
```

3 Záver

Prvým cieľom bakalárskej práce bolo zostrojiť telo kvadrokoptéry a vybrať vhodné súčiastky. Model rámu bol stiahnutý z internetu a jednotlivé diely rámu kvadrokoptéry boli vytlačené na 3D tlačiarni. Následne boli vybrané a kúpené vhodné súčiastky, ktoré boli namontované a prepojené. Tiež bolo vytlačené puzdro pre pripevnenie Raspberry Pi a MikroTik. Riadiaca jednotka kvadrokoptéry bola skalibrovaná pomocou počítača cez USB a tým pripravená na skúšobný let.

Ďalšou časťou bola inštalácia operačného systému FlytOS na zariadenie Raspberry Pi, ktoré ovláda riadiacu jednotkou Pixhawk. Komunikácia medzi riadiacou jednotkou a Raspberry Pi bola zaistená prostredníctvom vlastnoručne vyrobeného spoja. Počítač bol pripojený na Raspberry Pi prostredníctvom jeho Access Point. Vďaka programu Qgroundstation bolo možné nastaviť parametre riadiacej jednotky Pixhawk tak, aby správne komunikovala s FlytOS a tým umožnila sledovať aktuálne dianie na riadiacej jednotke. Konkrétne náklon riadiacej jednotky a jej aktuálnu polohu po pripojení GPS modulu. Aby FlytOS prevzalo úplnú kontrolu nad kvadrokoptérou bolo potrebné nastaviť módy letu. V jednom móde bola kvadrokoptéra ovládaná prostredníctvom FlytOS a v druhom cez diaľkový ovládač.

Počas prvého letu bola kvadrokoptéra ovládaná diaľkovým ovládačom ktorý dopadol úspešne a počas ďalšieho letu bola riadiaca jednotka preťažená kvôli napájaniu Raspberry Pi čo spôsobilo haváriu. Tento problém bol vyriešený samostatným napájaním Raspberry Pi.

Ďalším cieľom bakalárskej práce bolo vytvoriť simuláciu automatizovaného letu kvadrokoptéry v prostredí FlytSIM. Simulácia bola vytvorená v programovacom jazyku Python. Let pozostával zo vzletu a preletu nad dvomi bodmi na základe GPS súradníc. Počas letu nad zadanými bodmi sa kvadrokoptéra natočí na vzletové miesto a vráti sa na miesto vzletu. Bolo potrebné nastaviť parametre letu v simulácii tak, aby kvadrokoptéra letela správnym smerom, správne natočená a v určitej výške. Táto simulácia bola úspešne otestovaná a kvadrokoptéra sa správala tak, ako mala zadané.

Dôležitou časťou je, aby počas celého letu kvadrokoptéra udržiavala spojenie so základovou stanicou. Z tohto dôvodu bol vytvorený algoritmus, do ktorého boli zadané súradnice letu a súradnice základovej stanice a jeho výstupom bol kód automatizovaného letu pre FlytOS, kde v každom bode bol vypočítaný uhol natočenia tak, aby vždy bolo udržané spojenie medzi anténami. Výstup tohoto algoritmu bol overený vo FlytSIM, ktorý dopadol úspešne. Na kvadrokoptére bolo použité zariadenie Mikrotik wAP 60G pre komunikáciu.

Zoznam symbolov, veličín a skratiek

A	Ampér
AP	Access point
API	Application programming interface
APM	ArduPilot Mega
C	Statický vybíjací prúd
CW	ClockWise
CCW	Counter ClockWise
FlytOS	Flyt Operating System
GCS	Ground Control Stations
GPIO	General Purpose Input and Output
IoD	Internet of Drones
KV	Revolutions Per Minut/Volt - počet otáčok za minútu na 1 Volt
Li-po	Lithium Polymer
LiDAR	Light Detec-tion And Ranging
MAVLink	Micro Air Vehicle Link
NFZ	No-Fly Zones
OS	Operating system
SDK	Software development kit
SoC	system on chip
RPA	Remotely piloted aircraft
RTOS	Real-Time Operating System
UAV	Unmanned Aerial Vehicle
UA	Unmanned aircraft
UAS	Unmanned Aircraft System
kg	Kilogram
g	Gram
mm	Milimeter
GHz	GigaHertz
mAh	Miliampér hodina
ROS	Robot Operating System

Literatúra

[1] D. Kyjovský, "Závodní minikvadkoptéry, přehled základních pojmů a úvod do problematiky," Česka republika, 2016. [Online]. Dostupné z: http://www.droneweb.cz/konstrukce/item/54-zavodni-kvadrokoptery-terminy-konstrukce

[2] R. Čorba *et al.*, "Realizace létajícího prostředku," 2016.

[3] "VELKÝ PRŮVODCE: Základy pro stavbu dronu!" Česká republika, 2018. [Online]. Dostupné z: https://svetdronu.net/velky-pruvodce-zaklady-pro-stavbu-dronu/

[4] "Ako vybrať dron," Česká republika, 2018. [Online]. Dostupné z: https://www.covybrat.sk/ako-vybrat-dron/

[5] "Motory v multikoptérach," Česká republika, 2015. [Online]. Dostupné z: https://www.rcportal.sk/motory-v-multikopterach-c307?fbclid=IwAR30VfOdIGLS5aRW9bkAFTLwiabEo9yjAtgKA9ZOHgdlWxt1YC92KizRZ8w

[6] "Dimenzování komponent pohonů copterů a letadel," Česká republika, 2014. [Online]. Dostupné z: http://www.fpvguru.cz/dimenzovani-komponent-pohonu-copteru-a-letadel.html

[7] "Jak se vyznat v označení vrtulí," Česká republika, 2018. [Online]. Dostupné z: https://www.rc-zoom.cz/jak-se-vyznat-v-oznaceni-vrtuli/

[8] "Pixhawk Series," Amerika, 2018. [Online]. Dostupné z: https://docs.px4.io/en/flight_controller/pixhawk_series.html

[9] L. Feng and Q. Fangchao, "Research on the Hardware Structure Characteristics and EKF Filtering Algorithm of the Autopilot PIXHAWK," pp. 228–231, 2016.

[10] "ArduPilot," Amerika, 2016. [Online]. Dostupné z: http://ardupilot.org/ardupilot/

[11] "SOFTWARE," Amerika, 2016. [Online]. Dostupné z: http://ardupilot.org/about

[12] "FlytOS Architecture," Amerika, 2018. [Online]. Dostupné z: https://flytbase.com/flytos/

[13] "FlytOS," Amerika, 2016. [Online]. Dostupné z: http://ardupilot.org/dev/docs/flytos.html

[14] “Flytos:why,what,who?” Amerika, 2016. [Online]. Dostupné z: https://blogs.flytbase.com/flytos-why-what-who/

[15] M. Horák, “Raspberry pi: Co to vlastně je,” Česká republika, 2012. [Online]. Dostupné z: http://www.abclinuxu.cz/clanky/raspberry-pi-co-to-vlastne-je

[16] “Raspberry Pi,” Česká republika, 2018. [Online]. Dostupné z: https://www.root.cz/n/raspberry-pi/

[17] “Choosing a Ground Station,” Amerika, 2016. [Online]. Dostupné z: http://ardupilot.org/copter/docs/common-choosing-a-ground-station.html

[18] S. Romaniuk, Z. Gosiewski, and L. Ambroziak, “A Ground Control Station for the UAV Flight Simulator,” *Acta Mechanica et Automatica*, vol. 10, 03 2016.

[19] S. Ali, A. Sardar, K. K. Patra, and R. Barua, “Development of low cost portable ARM processor based GCS data interface module for range sensors network,” pp. 76–79, 2015.

[20] C. Atencia and D. Camacho, “Extending QGroundControl for Automated Mission Planning of UAVs,” *Sensors*, vol. 2018, p. 23, 2018.

[21] “QGroundControl User Guide,” Amerika, 2017. [Online]. Dostupné z: https://docs.qgroundcontrol.com/en/

[22] “FlytSim,” Amerika, 2017. [Online]. Dostupné z: http://docs.flytbase.com/docs/FlytSim/docker/setup.html

Zoznam príloh

A Kód Simulovaného letu vo FlytSIM v jazyku Python

```
1  import time
   from flyt_python import api
3
   drone = api.navigation(timeout=120000)
5  #Vytvorenie instancie.

7  #Prikaz pre uspanie drona na 3 sekundy aby sa jeho rozhranie spravne
       nacitalo.
   time.sleep(3)
9
   #Prikaz pre vypis v konzole aby bolo jasne ktora cast kodu sa vykonava.
11 print 'Vzlet'

13 #Prikaz Pre vzlet UAV do vysky 5 m
   drone.take_off(5.0)
15
   print 'presun na 1 suradnicu'
17 #Prikaz pre presun drona vo formate (zemepisna vyska, sirka, vyska letu
       v metroch,natocenie smeru letu, potvrdenie ze sa moze dron otocit
       pocas letu na zadanu ssradnicu).
   drone.position_set_global(49.22912, 16.57343, 10, yaw=-2.3065,
       yaw_valid=True)
19
   print 'presun na 2 suradnicu'
21 drone.position_set_global(49.22875,16.57385, 5, yaw=-2.3065, yaw_valid=
       True)

23 print 'presun na 3 suradnicu'
   drone.position_set_global(49.22856, 16.57283, 15, yaw=-2.15, yaw_valid=
       True)
25
   print 'Pristatie'
27 #Prikaz pre pristanie drona.
   drone.land(async=False)
29
   #Prikaz pre zastavenie motorov drona.
31 drone.disconnect()
```

B Kód letu s medzisúradnicami vo FlytSIM v jazyku Python

```
import time
from flyt_python import api
drone = api.navigation(timeout=120000)
time.sleep(3.0)
drone.take_off(5.0)
drone.position_set_global(49.228805, 16.572634, 10.00, yaw = -4.
                                    324983, yaw_valid=True)
time.sleep(3)
drone.position_set_global(49.228833, 16.572734, 10.00, yaw = -4.
                                    203078, yaw_valid=True)
drone.position_set_global(49.228861, 16.572833, 10.00, yaw = -4.
                                    053354, yaw_valid=True)
drone.position_set_global(49.228875, 16.572883, 10.00, yaw = -3.
                                    967374, yaw_valid=True)
drone.position_set_global(49.228889, 16.572933, 10.00, yaw = -3.
                                    874295, yaw_valid=True)
drone.position_set_global(49.228903, 16.572983, 10.00, yaw = -3.
                                    774999, yaw_valid=True)
drone.position_set_global(49.228917, 16.573033, 10.00, yaw = -3.
                                    670969, yaw_valid=True)
drone.position_set_global(49.228932, 16.573083, 10.00, yaw = -3.
                                    564226, yaw_valid=True)
drone.position_set_global(49.228946, 16.573132, 10.00, yaw = -3.
                                    457134, yaw_valid=True)
drone.position_set_global(49.228960, 16.573182, 10.00, yaw = -3.
                                    352106, yaw_valid=True)
drone.position_set_global(49.228974, 16.573232, 10.00, yaw = -3.
                                    251287, yaw_valid=True)
drone.position_set_global(49.228988, 16.573282, 10.00, yaw = -3.
                                    156323, yaw_valid=True)
drone.position_set_global(49.229016, 16.573382, 10.00, yaw = -2.
                                    987600, yaw_valid=True)
drone.position_set_global(49.229040, 16.573467, 15.00, yaw = -2.
                                    866498, yaw_valid=True)
time.sleep(4)
drone.land(async=False)
drone.disconnect()
```

C Algoritmus výpočtu uhlu natočenia a medzisúradníc v jazyku Python

```
import math
# ****************FUNKCIE*******************
def vypocet(bs,bd,ys,yd):
    a = (yd - bd) * d
    b = (ys - bs) * s
    c = math.acos((math.sin(math.radians(bs)) * math.sin(math.radians(
    ys))) + (math.cos(math.radians(bs)) * math.cos(math.radians(ys)) *
    math.cos(math.radians(bd - yd)))) * 6371
    alfa = math.degrees(math.atan(b / a))
    if (a > 0 and b > 0): # 1 kvadrant
        alfa= alfa+0
    elif (a < 0 and b > 0): # 2 kv
        alfa = alfa + 180
    elif (a < 0 and b < 0): # 3 kv
        alfa = alfa + 180
    elif (a > 0 and b < 0): # 4 kv
        alfa = alfa + 360
    else:
        print('chyba pri vypocte trojuholnika a urcovani kvadrantu')
    return (alfa,a,b,c)
def vypocet2(bs,bd,ys,yd):
    a = (yd - bd) * d
    b = (ys - bs) * s
    c = math.acos((math.sin(math.radians(bs)) * math.sin(math.radians(
    ys))) + (math.cos(math.radians(bs)) * math.cos(math.radians(ys)) *
    math.cos(math.radians(bd - yd)))) * 6371
    alfa = math.degrees(math.atan(a / b))
    alfa = math.sqrt(alfa ** 2)
    if (a > 0 and b > 0): # 1 kvadrant
        alfa= 180 + (90 - alfa)
    elif (a < 0 and b > 0): # 2 kv
        alfa = 270 + alfa
    elif (a < 0 and b < 0): # 3 kv
        alfa = 90 - alfa
    elif (a > 0 and b < 0): # 4 kv
        alfa = alfa + 90
    else:
        print('chyba pri vypocte trojuholnika a urcovani kvadrantu')
    return (alfa,a,b,c)

def rozdieluhlov(x,y):
    if (x-y <= 180):
```

```
39          alfa = x-y
        elif (x-y > 180):
41          alfa = 360-x+y
        else:
43          print ('Podmienka rozdielu uhlov nebola dodrzana')
        return alfa
45
def rozdielpodmienka(xalfa,yalfa):
47      if (xalfa>yalfa):
            alfa = rozdieluhlov(xalfa,yalfa)
49      elif (xalfa<yalfa):
            alfa = rozdieluhlov(yalfa,xalfa)
51      else:
            alfa = xalfa
53      return alfa
################################################################################
55
# Zadavanie vstupnych premennych
57 wp = int(input("Pocet bodov preletu: "))
bs = 49.228630 #float(input("zemepisna sirka BS: "))
59 bd = 16.573290 #float(input("Zemepisna dlzka BS: "))
u = int(input("odchylka v stupnoch medzi bodmi letu: "))
61 #Vseobecne premenne
s = 111190 # 1 stupen zemepisnej sirky 6371 / 360 * 2 * Pi = 111.19
63 d = 40075000 * math.cos(math.radians(bs)) / 360 #1 stupen zemepisnej
    dlzky
i = 0
65 # Dopredu vytvorene polia kore budu hlavne vyuzivane
wps = [] #zadane zemepisne sirky zo vstupu
67 wpd = []#zadane zemepisne dlzky zo vstupu
wpvysky = [] #zadane vysky zo vstupu
69 wpcakania = [] #zadana doba cakania zo vstupu
puhol = [0, 0, 0] #pomocne pole do ktoreho sa ukladaju mezi vypocty
71 palfa = [] #vypocitane uhly
ps = [] # vsetky vysledne zemepisne sirky
73 pd = []# vsetky vysledne zemepisne dlzky
pvysky = []  # vysledne pole vysky letu pre kazdu suradnicu
75 pcakania = [] # vysledne pole cakania pre kazdu suradnicu
uholnatoceniadronu = [] # vypocitane natocenie drona pre kazdu
    suradnicu na BS
77
# Naplnenie polia so suradnicami letu vyskou letu a dobou cakania na
    suradnici
79 while i < wp:
    wps.append(float(input("zemepisna sirka bodu cislo {}: " .format(i
    + 1))))
```

```
81      wpd.append(float(input("zemepisna dlzka bodu cislo {}: " .format(i
        + 1))))
        wpvysky.append(float(input("vysku letu v metroch {}: " .format(i +
        1))))
83      wpcakania.append(float(input("doba cakania na danom bode {}: ".
        format(i + 1))))
        i = i + 1
85  print(wps, " ", wpd, " ", wpvysky, " ", wpcakania)
    i = 0
87  wpalfa = []

89  # Vlozenie vstupnych udajov do vystupnych poli na dalsie spracovanie
        cyklus s opakovanim podla toho kolko suradnic bolo zadanych
    while i + 1 < wp:
91      f = 1
        #Naplnenie prvkov prvymi udajmi (konkretne zadavane cisla z
        prikazoveho riadku)
93      pcakania.append(wpcakania[i])
        pvysky.append(wpvysky[i])
95      ps.append(wps[i])
        pd.append(wpd[i])
97      xalfa, xa, xb, xc = vypocet(bs, bd, wps[i], wpd[i])
        wpalfa.append(xalfa)
99      palfa.append(xalfa)

101     puhol[0] = xalfa
        ys = wps[i+1]
103     yd = wpd[i+1]
        xs = wps[i]
105     xd = wpd[i]
    # Podmienka aby sa suradnice letu nenachdzali na rovnakej dlzke alebo
        sirke ako BS inak vypocet nebude fungovat
107     if (bs != ys) and (bd != yd) and (bs != xs) and (bd != xd):
            yalfa, ya, yb, yc = vypocet(bs, bd, ys, yd)
109         xalfa, xa, xb, xc = vypocet(bs, bd, xs, xd)
        else:
111         print('bod nesmie byt na rovnakej suradnici ako BS zruste a
        opakujte ')
        alfa = rozdielpodmienka(xalfa, yalfa)
113     wpalfa.append(yalfa)
        i += 1
115     # porovnanie ak je ruzdiel uhlov vecsia ako zadana citlivost
        vypocitaju sa mezibody letu
        if (alfa > u):
117         #vypocet kroku zemepisnej dlzky a sirky
            zs = (ys - xs)/(alfa/u)
119         zd = (yd - xd)/(alfa/u)
```

```
        sirka = xs+zs
        dlzka = xd+zd
            #pomocne premenne
        cislo=0
            #cyklus pre vypocet pola bodov suradnic medzi bodom odkial leti a kam leti
        while(f == 1):
            puhol[1], nic1, nic2, nic3 = vypocet(bs, bd, sirka, dlzka)
            puhol[2] = rozdielpodmienka(puhol[0], puhol[1])
            pocetopakovani=0
                #podmienka ak krok je vecsi ako citlivost krok sa skrati s kazdym opakovanim
            while (puhol[2] > u):
                pocetopakovani = pocetopakovani+1
                sirka = sirka - (zs - (zs/(2*pocetopakovani)))
                dlzka = dlzka - (zd - (zd/(2*pocetopakovani)))
                puhol[1], nic1, nic2, nic3 = vypocet(bs, bd, sirka, dlzka)
                puhol[2] = rozdielpodmienka(puhol[0], puhol[1])

            cislo = rozdielpodmienka(xalfa, puhol[1])
            if cislo < alfa:
                    #Naplnenie pola vypocitanym krokom
                palfa.append(puhol[1])
                puhol[0] = puhol[1]
                ps.append(sirka)
                pd.append(dlzka)
                pvysky.append(wpvysky[i-1])
                pcakania.append(0)
                    #nove suradnice
                sirka = sirka+zs
                dlzka = dlzka+zd
            else:
                f = 0
#Naplnenie poli poslednym prvkom
palfa.append(yalfa)
ps.append(ys)
pd.append(yd)
pvysky.append(wpvysky[i])
pcakania.append(wpcakania[i])
pocetopakovani = 0
# Vypocitanie natocenia samotneho drona
while (pocetopakovani+1 <= len(pd)):
    f, nic1, nic2, nic3 = vypocet2(bs, bd, ps[pocetopakovani], pd[pocetopakovani])
    uholnatoceniadronu.append(f)
    pocetopakovani = pocetopakovani + 1
```

```
# kontrolny vypis vsetkych udajov a velkosti poli
print('dlzka', pd, '\n')
print('sirka', ps, '\n')
print('vyska letu', pvysky, '\n')
print('uhol natocenia zakladne', palfa, '\n')
print('uhol natocenia dronu', uholnatoceniadronu, '\n')
print('doba cakania na suradnici', pcakania, '\n')
print(len(pd), len(ps), len(pvysky), len(palfa), len(uholnatoceniadronu
    ), len(pcakania))

############### METODA CEZ GLOBAL POSITION
i = 0
uholnatoceniadronurad = []
subor = open('flyGP.py', 'w')
subor.write('import time')
subor.write('\n')
subor.write('from flyt_python import api')
subor.write('\n')
subor.write('drone = api.navigation(timeout=120000)')
subor.write('\n')
subor.write('time.sleep(3.0)')
subor.write('\n')
subor.write('drone.take_off(5.0)')
subor.write('\n')
i = 0
while i < len(ps):
    #prepocitanie uhlu lebo FlytOS nama nulu v pravo na jednotkovej
    kruznici ale hore a prepocet zo stupnou na radiany
    if uholnatoceniadronu[i] < 90:
        uholnatoceniadronurad.append((270 + uholnatoceniadronu[i]) *
    -0.017453292519943295)
    else:
        uholnatoceniadronurad.append((uholnatoceniadronu[i] - 90) *
    -0.017453292519943295)
    subor.write('drone.position_set_global(')
    subor.write("%f" % ps[i])
    subor.write(", ")
    subor.write("%f" % pd[i])
    subor.write(", ")
    subor.write("%.2f" % pvysky[i])
    subor.write(", yaw = ")
    subor.write("%f" % uholnatoceniadronurad[i])
    subor.write(", yaw_valid=True")
    subor.write(")")
    subor.write('\n')
    if pcakania[i] > 0:
        subor.write("time.sleep(")
```

```
            subor.write("%d" % pcakania[i])
207         subor.write(")")
            subor.write('\n')
209     i += 1
subor.write('\n')
211 subor.write('drone.land(async=False)')
subor.write('\n')
213 subor.write('drone.disconnect()')
```

text/kod.py